```
/*
+-----------------------------------------------------------------------------
-
 |    Copyright (c) 2002, 2003 © Hewlett-Packard Development Company, L.P.
 |    All rights reserved
 |
 |    Hewlett-Packard Confidential.
 |
 |    Top is a program designed to work with the virtual install disk
 |    HTML integration screen.  Currently with .NET, after an unassisted OS
 |    install the first time that the customer logs in, the integration
 |    screen is displayed but then is immediately covered up by the .NET
 |    "Manage Your Server" screen.
 |
 |    Top fixes this issue by loading after IE, finding IE's window handle,

 |    sets the "always on top" flag for the browser window, and then waits
 |    for the "Manage Your Server" screen to run and then turns off the
 |    "always on top flag" for normal behavior and sets the focus
accordingly.
 |
 |    Author: Rob Greenberg
 |
 |    Version: 1.005 - First external release
 |                     1.006 - Focus enhancements added and no longer matches
for MYS
 |                                window title to fix localization issues.
(Now looks
 |                                for MYS window title to be non-blank and
not contain
 |                                the string 'dll')

+-----------------------------------------------------------------------------
 */

/*-----------------------------------------------------------------------------
---
 |                                HEADER FILES

+-----------------------------------------------------------------------------
*/
#include <windows.h>
#include <tchar.h>
#include <stdio.h>
#include <assert.h>


BOOL CALLBACK EnumWindowsProc(HWND hwnd,      // handle to parent window
                              LPARAM lParam   // application-defined value
                             );

void myOutputDebugString (char *message);
void LogFocus(char *message);

HWND FoundhWnd;
HWND hHTMLWnd;
int state;

#define HTML_TIMELIMIT        5000
#define MYS_TIMELIMIT         60000
#define MYS_TITLE_TIMELIMIT   15000
#define LOOKING_FOR_HTML      0
#define LOOKING_FOR_MYS       1
#define LOOKING_FOR_MYS_TITLE 2

#define IE_CLASSNAME          "ieframe"
#define MYS_CLASSNAME         "html application host window class"
```

```
#define MYS_WINDOW_TITLE        "dll"

//#define DEBUGGING 1


/*
 ---------------------------------------------------------------------
 --

 --        Name:             WinMain

 --

 --  Description: Main entry point from Windows

 --

 --

 ---------------------------------------------------------------------
*/
int APIENTRY WinMain (HINSTANCE hInstance,
                      HINSTANCE hPrevInstance,
                      LPSTR     lpCmdLine,
                      int       nCmdShow)
{
    //-------------------------------------------------------------------
    //  Get the current system timer count in milliseconds
    //-------------------------------------------------------------------
    DWORD dwStart = GetTickCount();

    state = LOOKING_FOR_HTML;
    myOutputDebugString("Starting");

    //-------------------------------------------------------------------
    //  Loop until we find the IE window handle or our timelimit is
exceeded
    //-------------------------------------------------------------------
    while (1)
    {

//-------------------------------------------------------------------
        //  If TIMELIMIT milliseconds have elapsed, break out of the loop
and exit

//-------------------------------------------------------------------
        if (state == LOOKING_FOR_HTML)
           if( GetTickCount() - dwStart >= HTML_TIMELIMIT)
              break;

        if (state == LOOKING_FOR_MYS)
           if( GetTickCount() - dwStart >= MYS_TIMELIMIT)
              break;

        if (state == LOOKING_FOR_MYS_TITLE)
           if( GetTickCount() - dwStart >= MYS_TITLE_TIMELIMIT)
              break;


//-------------------------------------------------------------------
        //  Enumerate all open top-level Windows

//-------------------------------------------------------------------
        FoundhWnd = 0;
        EnumWindows(EnumWindowsProc, 0);
```

```
//-------------------------------------------------------------------
        //  If our hWnd variable is non-zero, we've found the hWnd we
wanted
//-------------------------------------------------------------------
        if (FoundhWnd != 0)
        {
            int ret;

            assert(state == LOOKING_FOR_HTML || state == LOOKING_FOR_MYS ||
                      state == LOOKING_FOR_MYS_TITLE);


//-------------------------------------------------------------------
            //  Force IE to be displayed and maximized and then set IE's
            //  "always on top" attribute w/o changing its window size or
position
//-------------------------------------------------------------------
            if (state == LOOKING_FOR_HTML)
            {
                myOutputDebugString("Found HTML Window");
                hHTMLWnd = FoundhWnd;                   // save for later use
                ShowWindow(hHTMLWnd, SW_SHOWMAXIMIZED);
                SetForegroundWindow(hHTMLWnd);
                ret = SetWindowPos(hHTMLWnd, HWND_TOPMOST, 0,0,0,0, SWP_NOSIZE
     | SWP_NOMOVE | SWP_SHOWWINDOW);
                LogFocus("Focus: initial focus after TOPMOST set");
                assert(ret);


//-------------------------------------------------------------------
                //  Reset for next pass -- looking for the Manage Your Server
screen
                //  reset the timer, the successful hWnd, and change state
variable
                //  Then continue in the while loop
//-------------------------------------------------------------------
                dwStart = GetTickCount();
                state = LOOKING_FOR_MYS;
                continue;
            }
            else
//-------------------------------------------------------------------
            //  If we found the "Manage Your Server" window
            //  wait for the "Manage Your Server" title bar to appear
            //  (this could be a second or even longer)
//-------------------------------------------------------------------
            if (state == LOOKING_FOR_MYS)
            {
//-------------------------------------------------------------------
                //  Reset for next pass -- looking for the MYS screen's window
title
                //  reset the timer, the successful hWnd, and change state
variable
                //  Then continue in the while loop
//-------------------------------------------------------------------
                myOutputDebugString("Found MYS Window");
                LogFocus("Focus: Initial found MYS Window");
                dwStart = GetTickCount();
                state = LOOKING_FOR_MYS_TITLE;
                continue;
            }
```

```
        else

//-------------------------------------------------------------------------
        //  If we found the "Manage Your Server" text in the MYS window
        //  wait a second for the screen to draw and then turn off the
        //  IE "always on top" attribute and force the focus to IE

//-------------------------------------------------------------------------
        if (state == LOOKING_FOR_MYS_TITLE)
        {
            myOutputDebugString("Found MYS Title, starting 1 sec sleep");
            Sleep(1000);


//-------------------------------------------------------------------------
            // If the HTML Window is still valid (not closed)

//-------------------------------------------------------------------------
            if (IsWindow(hHTMLWnd))
            {
                DWORD dwForegroundThread;
                DWORD dwIEThread;
                BOOL bRetCode;

                LogFocus("Focus: About to turn off topmost");


//-------------------------------------------------------------------------
                // Turn off the topmost attribute on the HTML Window

//-------------------------------------------------------------------------
                ret = SetWindowPos(hHTMLWnd, HWND_NOTOPMOST, 0,0,0,0,
SWP_NOSIZE | SWP_NOMOVE | SWP_SHOWWINDOW);
                assert(ret);

                LogFocus("Focus: topmost off");


//-------------------------------------------------------------------------
                // Set the focus to the HTML screen.  AttachThreadInput on
                // IE if necessary to allow the setfocus to work correctly

//-------------------------------------------------------------------------

dwForegroundThread=GetWindowThreadProcessId(GetForegroundWindow(),NULL);
                dwIEThread=GetWindowThreadProcessId(hHTMLWnd,NULL);

                 if (dwForegroundThread != dwIEThread)
                 {
                    bRetCode = AttachThreadInput(dwForegroundThread,
dwIEThread, TRUE);
                    assert(bRetCode);

                    bRetCode = SetForegroundWindow(hHTMLWnd);
                    assert(bRetCode!=0);

                    LogFocus("Focus: After attach threadinput, focus on
HTML again");

                    // Allow time for focus change and activation
                    Sleep(500);

                    bRetCode = AttachThreadInput(dwForegroundThread,
dwIEThread, FALSE);
                    assert(bRetCode);
                 }
                else
                 {
```

```
                    bRetCode = SetForegroundWindow(hHTMLWnd);
                    LogFocus("Focus: No threadinput attach needed; focus
still on HTML");
                    assert(bRetCode);
                 }
              }
              else
                 assert(FALSE);
           }
           break;      // we're ready to exit the program
      }
      else

//---------------------------------------------------------------------
      //  We didn't find the hWnd we were looking for, so sleep for a 5th
of
      //  a second before looping and re-enumerating the windows

//---------------------------------------------------------------------
      {
         myOutputDebugString("Sleeping");
         Sleep(200);
      }
   }

   myOutputDebugString("Exiting program");

   return 0;
}


/*

---------------------------------------------------------------------------
--
 --

 --        Name:             EnumWindowsProc

 --

 --  Description: Called from EnumWindows for each open top level window
 --               looking for a window classname that matches our
expectations
 --               for the program state that we're in.
 --
 --               When we find the window classname or window title that we
 --               expect, we return and signal the end of the enumeration.
 --

 -- Return Value: TRUE to continue enumeration, FALSE to stop enumeration

 --


---------------------------------------------------------------------------
--
*/
BOOL CALLBACK EnumWindowsProc(HWND hwnd,      // handle to parent window
                              LPARAM lParam   // application-defined value
                             )
{
    char classname[250];
    char windowtitle[250];

    memset(classname, 0, sizeof(classname)-1);
    memset(windowtitle, 0, sizeof(windowtitle)-1);
```

```
    //---------------------------------------------------------------------
    //  hwnd should always be non-null but check just in case...
    //---------------------------------------------------------------------
    if (hwnd != NULL)
    {

//---------------------------------------------------------------------
        //  Get the window class name for this window and make it lower
case
        //  in a fashion compatible with multibyte or unicode strings

//---------------------------------------------------------------------
        GetClassName(hwnd, classname, sizeof(classname)-10);
        _tcslwr(classname);


//---------------------------------------------------------------------
        //  If we've found IE's or Manage Your Server's classname string in
        //  the window, save the hWnd and return FALSE to stop the
enumeration

//---------------------------------------------------------------------
        if (
              (state == LOOKING_FOR_HTML &&
_tcsstr(classname,IE_CLASSNAME)) ||
              (state == LOOKING_FOR_MYS &&
_tcsstr(classname,MYS_CLASSNAME))
            )
        {
           FoundhWnd = hwnd;
           return FALSE;     // stop the enumeration
        }


//---------------------------------------------------------------------
        //  If we're looking for the Manage Your Server window title and
find
        //  the correct MYS classname string, check the window title.  If
we
        //  find a match, save the hWnd and return FALSE to stop the
enumeration

//---------------------------------------------------------------------
        if (state == LOOKING_FOR_MYS_TITLE)
        {
#ifdef DEBUGGING
            char scratch[1000];

            sprintf(scratch, "Found classname: %s", classname);
            myOutputDebugString(scratch);
#endif


//---------------------------------------------------------------------
            //  If the classname matches keep looking for our desired MYS
window

//---------------------------------------------------------------------
            if (_tcsstr(classname,MYS_CLASSNAME))
            {
               long myLong;

               myLong = GetWindowLong(hwnd, GWL_EXSTYLE);

#ifdef DEBUGGING
               sprintf(scratch, "Window extended style APPWINDOW is: %ld",
myLong & WS_EX_APPWINDOW);
```

```
                myOutputDebugString(scratch);
#endif

//---------------------------------------------------------------------
            //  If this window has an entry in the taskbar (ie, it's
visible),
            //  we've found the MYS window that we want

//---------------------------------------------------------------------
            if (myLong & WS_EX_APPWINDOW)
            {
                  GetWindowText(hwnd, windowtitle, sizeof(windowtitle)-10);
                  _tcslwr(windowtitle);

#ifdef DEBUGGING
                  sprintf(scratch, "Found window title: %s", windowtitle);
                  myOutputDebugString(scratch);
#endif

//---------------------------------------------------------------------
                  //  If the window has a title and it doesn't contain the
string "dll"
                  //  then the real MYS window title has been set and we
have a match
                  //
                  //  Possible window titles include:
                  //  1.  No window title
                  //  2.  The path of the DLL that MYS was loaded from
                  //  3.  The ultimate window title for the MYS window
                  //
                  //  This program may find the title in any of the above
states but
                  //  ulimately, the window title will end up in state 3
and this is
                  //  the state that the program tries to identify

//---------------------------------------------------------------------
                  if (
                        _tcslen(windowtitle) &&
                        (_tcsstr(windowtitle,MYS_WINDOW_TITLE) == 0)
                     )
                  {
                     FoundhWnd = hwnd;
                     myOutputDebugString("Matched window class and title");
                     return FALSE;      // stop the enumeration
                  }
                  else
                     myOutputDebugString("Window class and title DID NOT
match");
               }
            }
            myOutputDebugString("Returned from looking for mys title
without match");
         }
      }
      return TRUE;            // continue the enumeration
   }

/*

---------------------------------------------------------------------------
--
 --

 --         Name: myOutputDebugString
```

```
 --
 --  Description: Debugging support to debugger and file
 --
 -- Return Value: None
 --

--------------------------------------------------------------------------
--
*/
void myOutputDebugString (char *message)
{
#ifdef DEBUGGING
     char message_buff[500];

     FILE *fp;

     fp = fopen("c:\\debug.log", "a");

     sprintf(message_buff, "%s %d %d   |   ", message, state,
GetTickCount());

     if (fp)
     {
         fputs(message_buff, fp);
         putc('\n', fp);
         fclose(fp);
      }

      OutputDebugString(message_buff);
#endif
}


void LogFocus(char *message)
{
#ifdef DEBUGGING
     char classname[250];
     char windowtitle[250];
     char scratch[600];
     HWND hWnd1;

     hWnd1 = GetForegroundWindow();
     GetClassName(hWnd1, classname, sizeof(classname)-10);
     GetWindowText(hWnd1, windowtitle, sizeof(windowtitle)-10);

     myOutputDebugString(message);
     sprintf(scratch, "classname: %s", classname);
     myOutputDebugString(scratch);
     sprintf(scratch, "windowtitle: %s", windowtitle);
     myOutputDebugString(scratch);
     myOutputDebugString(" ");
#endif
}
```